# Python 大作业——人物信息检索系统

## 一、 工程简介

Python 对网络爬虫的支持非常好。这次工程尝试运用 Python 的 beautifulsoup 库来建立爬虫，抓取维基百科里 NBA 与 MLB 历史球员的全部信息。储存在本地后，运用 django 架构建设本地服务器，通过检索系统搜索关键词，将球员信息显示在网页上。

## 二、 程序结构

这个任务检索系统一共分为两个部分，分别是信息收集和网页实现。第一部分中，我首先在 NBA 历史球员和 MLB 历史球员页面，把全部球员的个人界面的网址扒下来，生成 NBAplayerlist 和 MLBplayerlist 两个文件。再写了根据这两个文件的内容，运用 beautifulsoup，依照 wiki 不同界面的 infobox 的不同格式，分别写出了几个爬虫程序将他们的 infobox 内容和头像储存在本地文件中。第二部分我用了 django 的结构，运用两个 html 文件分别显示搜索界面和结果界面。

## 三、 代码实现

1、 爬虫

首先是 url 的扒取，我发现历史球员界面中每个球员都有自己的个人页面，其 url 就在<a>标签中。得到 urls 后，我遍历读取 NBAplayerlist 中的所有 url，进入网页中寻找 infobox 并依据一定格式储存在本地中。

```
import urllib2
import urllib
from urllib2 import HTTPError
from bs4 import BeautifulSoup
import cPickle as pickle
import re

url = 'https://en.wikipedia.org/wiki/List_of_Major_League
user_agent = 'Mozilla/4.0 (compatible; MSIE 5.5; Windows
headers = {'User-Agent': user_agent}
values = {'q': 'python'}
data = urllib.urlencode(values)
req = urllib2.Request(url, data, headers)
response = urllib2.urlopen(req)
the_page = response.read()
soup = BeautifulSoup(the_page, 'lxml')
playerlist = []
num = 0
p = soup.find_all('tr')
boo = 0
for j in p:
    if boo == 0:
        boo = 1
        continue
    h = j.td.a['href']
    print h
    temp_url = "https://en.wikipedia.org" + h
```

```
    playerlist.append(temp_url)
    num = num + 1
    print num
inp = open('MLBplayerslist', 'r')
players = pickle.load(inp)
players.extend(playerlist)
inp.close()

output = open('MLBplayerslist', 'w')
pickle.dump(players, output)
output.close()
```

```
Listed weight$   235lb (107kg);
1969-1970$       Milwaukee Bucks;
Career highlights and awards$    Third-teamAll-
American-NABC(1968); Big Eight Player of the Year
(1968); No. 35retired by Iowa State;
1970-1972$       Seattle SuperSonics;
Assists$         601 (1.2 apg);
Nationality$     American;
Rebounds$        4,065 (8.0 rpg);
Listed height$   6ft 9in (2.06m);
1972-1975$       Houston Rockets;
NBA draft$       1968/ Round: 1 / Pick: 5th overall;
Selected by theCincinnati Royals;
High school$     John Jay(New York City, New York);
Position$        Power forward/Center;
name$   Zaid Abdul-Aziz
1978$    Houston Rockets;
1976$    Seattle SuperSonics;
Number$  21, 16, 35, 6, 54, 27;
1968-1969$       Cincinnati Royals;
Born$    (1946-04-07)April 7, 1946(age71)Brooklyn,
New York;
Points$  4,557 (9.0 ppg);
College$         Iowa State(1965-1968);
1976-1977$       Buffalo Braves;
Playing career$  1968-1978;
```

2、 倒排列表

```
dic = {'league': {},
       'name': {},
       'born': {},
       'number': {},
       'nba draft': {},
       'nationality': {},
       'position': {},
       'education': {},
       'career highlights and awards': {},
       'died': {},
       'allpro': {},
       'last mlb appearance': {},
       'teams': {},
       'mlb debut': {},
       }
playerlist=[]
```

在建立倒排列表时，我将读取的个人信息依照排序储存进 playerlist 中，首先创建一个字典，把本地文件中$前的信息为属性，作为字典的键，后面的信息作为字典的值存储，然后再 append 到 playerlist 中。我创建了一个字典 dic 储存不同属性的不同单词的倒排列表。倒排列表以 set 形式储存，以去重。最后将 playerlist 和 dic 通过 pickle 存储在本地文件中。

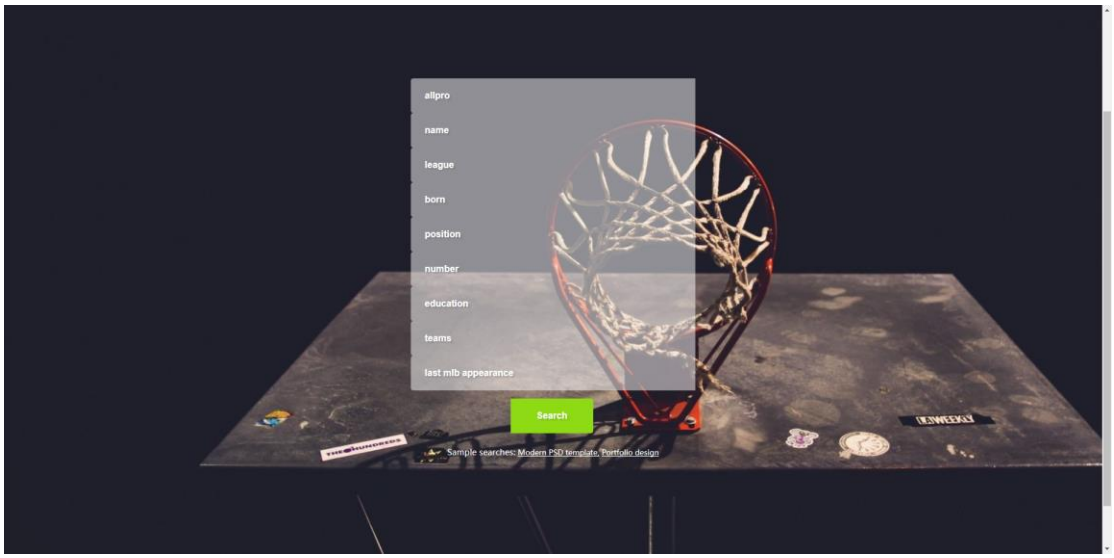
3、 搜索

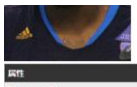
我运用 django 的虚拟 url，分别映射到 search 页面和 result 页面。在打开 search 页面时，读取本地的 dic 和 playerlist 作为 settings 中的变量存储。在输入查询关键字后表单用 get 方法，返回 result 函数。根据 request.GET 的参数信息，我从 dic 中提取倒排列表，若有多个关键词则将不同倒排列表的 set 进行与操作，得到共有的 player 编号，作为参数打开 result 页面。

4、 界面显示

我运用了模板方法，对 result 的参数信息循环遍历，每个运动员就绘制一个 显示其信息。运用了分页，实现最多 5 人一页。在制作界面的时候，我借鉴修改了网上的模板，实现了界面美化。

```
{% for i in contacts %}
    <div id="page-wrap" style="margin:50px 0 0">
        <h2>{{ i.name }}</h2>
        {% if 'image' in i %}
            {% load staticfiles %}
            <img src="{% static i.image %}">
        {% endif %}
        <table>
            <thead>
            <tr>
                <th>属性</th>
                <th>内容</th>
            </tr>
            </thead>
            <tbody>
            {% for j,k in i.items %}
                {% if j != 'image' and j != 'name' %}
                <tr>
                    <td>{{ j }}</td>
                    <td>{{ k }}</td>
                </tr>
                {% endif %}
            {% endfor %}
            </tbody>
        </table>
    </div>
{% endfor %}
<br><br>
```



| 属性 | 内容 |
| --- | --- |
| listed weight | 220lb (100kg); |
| listed height | 6ft 5in (1.96m); |
| born | (1994-07-08)july 8, 1994(age23)atlanta, georgia; |
| nba draft | 2014/ round: 1 / pick: 22nd overall: selected by thememphis grizzlies; |
| high school | oak hill academy(mouth of wilson, virginia); |
| college | ucla(2012-2014); |
| 2014–2015 | →iowa energy; |
| 2014–2016 | memphis grizzlies; |
| nationality | american; |
| position | shooting guard; |
| career highlights and awards | first-teamall-pac-12(2014); |
| playing career | 2014-present; |

Michael Jordan

| 属性 | 内容 |
| --- | --- |
| league | nba; |

## 四、 反思展望

这次大作业虽然基本完成，但是这个程序仍有许多可以完善的地方。首先是页面的 ui 不够仍然美观，友好，结果界面可以更好看一点。

其次我在 django 架构方向上不够熟练，尝试用 post 方法显示结果，但失败了。而且我的搜索结果也是存储在 settings 文件中，没法实现多个页面同时检索。初步想法是把搜索请求放在一个 list 中，不同用户使用不同编号的搜索结果。